Class09 Mini Project

Pierce Ford

10/27/2021

Unsupervised Learning Analysis of Human Breast Cancer Cells

```
#Read in the data file
fna.data <- "WisconsinCancer.csv"

#Convert data to data frame
wisc.df <- read.csv(fna.data, row.names=1)

#View the data to determine if the structure is as expected
str(wisc.df)
```

```
## ’data.frame':    569 obs. of  32 variables:
##  $ diagnosis              : chr  "M" "M" "M" "M" ...
##  $ radius_mean            : num  18 20.6 19.7 11.4 20.3 ...
##  $ texture_mean           : num  10.4 17.8 21.2 20.4 14.3 ...
##  $ perimeter_mean         : num  122.8 132.9 130 77.6 135.1 ...
##  $ area_mean              : num  1001 1326 1203 386 1297 ...
##  $ smoothness_mean        : num  0.1184 0.0847 0.1096 0.1425 0.1003 ...
##  $ compactness_mean       : num  0.2776 0.0786 0.1599 0.2839 0.1328 ...
##  $ concavity_mean         : num  0.3001 0.0869 0.1974 0.2414 0.198 ...
##  $ concave.points_mean    : num  0.1471 0.0702 0.1279 0.1052 0.1043 ...
##  $ symmetry_mean          : num  0.242 0.181 0.207 0.26 0.181 ...
##  $ fractal_dimension_mean : num  0.0787 0.0567 0.06 0.0974 0.0588 ...
##  $ radius_se              : num  1.095 0.543 0.746 0.496 0.757 ...
##  $ texture_se             : num  0.905 0.734 0.787 1.156 0.781 ...
##  $ perimeter_se           : num  8.59 3.4 4.58 3.44 5.44 ...
##  $ area_se                : num  153.4 74.1 94 27.2 94.4 ...
##  $ smoothness_se          : num  0.0064 0.00522 0.00615 0.00911 0.01149 ...
##  $ compactness_se         : num  0.049 0.0131 0.0401 0.0746 0.0246 ...
##  $ concavity_se           : num  0.0537 0.0186 0.0383 0.0566 0.0569 ...
##  $ concave.points_se      : num  0.0159 0.0134 0.0206 0.0187 0.0188 ...
##  $ symmetry_se            : num  0.03 0.0139 0.0225 0.0596 0.0176 ...
##  $ fractal_dimension_se   : num  0.00619 0.00353 0.00457 0.00921 0.00511 ...
##  $ radius_worst           : num  25.4 25 23.6 14.9 22.5 ...
##  $ texture_worst          : num  17.3 23.4 25.5 26.5 16.7 ...
##  $ perimeter_worst        : num  184.6 158.8 152.5 98.9 152.2 ...
##  $ area_worst             : num  2019 1956 1709 568 1575 ...
##  $ smoothness_worst       : num  0.162 0.124 0.144 0.21 0.137 ...
##  $ compactness_worst      : num  0.666 0.187 0.424 0.866 0.205 ...
##  $ concavity_worst        : num  0.712 0.242 0.45 0.687 0.4 ...
##  $ concave.points_worst   : num  0.265 0.186 0.243 0.258 0.163 ...
##  $ symmetry_worst         : num  0.46 0.275 0.361 0.664 0.236 ...
##  $ fractal_dimension_worst: num  0.1189 0.089 0.0876 0.173 0.0768 ...
##  $ X                      : logi  NA NA NA NA NA NA ...
```

```
head(wisc.df)
```

```
##          diagnosis radius_mean texture_mean perimeter_mean area_mean
## 842302           M       17.99        10.38         122.80    1001.0
## 842517           M       20.57        17.77         132.90    1326.0
## 84300903         M       19.69        21.25         130.00    1203.0
## 84348301         M       11.42        20.38          77.58     386.1
## 84358402         M       20.29        14.34         135.10    1297.0
## 843786           M       12.45        15.70          82.57     477.1
##          smoothness_mean compactness_mean concavity_mean concave.points_mean
## 842302           0.11840          0.27760         0.3001             0.14710
## 842517           0.08474          0.07864         0.0869             0.07017
## 84300903         0.10960          0.15990         0.1974             0.12790
## 84348301         0.14250          0.28390         0.2414             0.10520
## 84358402         0.10030          0.13280         0.1980             0.10430
## 843786           0.12780          0.17000         0.1578             0.08089
##          symmetry_mean fractal_dimension_mean radius_se texture_se perimeter_se
## 842302          0.2419                0.07871    1.0950     0.9053        8.589
## 842517          0.1812                0.05667    0.5435     0.7339        3.398
## 84300903        0.2069                0.05999    0.7456     0.7869        4.585
## 84348301        0.2597                0.09744    0.4956     1.1560        3.445
## 84358402        0.1809                0.05883    0.7572     0.7813        5.438
## 843786          0.2087                0.07613    0.3345     0.8902        2.217
##          area_se smoothness_se compactness_se concavity_se concave.points_se
## 842302    153.40      0.006399        0.04904      0.05373           0.01587
## 842517     74.08      0.005225        0.01308      0.01860           0.01340
## 84300903   94.03      0.006150        0.04006      0.03832           0.02058
## 84348301   27.23      0.009110        0.07458      0.05661           0.01867
## 84358402   94.44      0.011490        0.02461      0.05688           0.01885
## 843786     27.19      0.007510        0.03345      0.03672           0.01137
##          symmetry_se fractal_dimension_se radius_worst texture_worst
## 842302       0.03003             0.006193        25.38         17.33
## 842517       0.01389             0.003532        24.99         23.41
## 84300903     0.02250             0.004571        23.57         25.53
## 84348301     0.05963             0.009208        14.91         26.50
## 84358402     0.01756             0.005115        22.54         16.67
## 843786       0.02165             0.005082        15.47         23.75
##          perimeter_worst area_worst smoothness_worst compactness_worst
## 842302            184.60     2019.0           0.1622            0.6656
## 842517            158.80     1956.0           0.1238            0.1866
## 84300903          152.50     1709.0           0.1444            0.4245
## 84348301           98.87      567.7           0.2098            0.8663
## 84358402          152.20     1575.0           0.1374            0.2050
## 843786            103.40      741.6           0.1791            0.5249
##          concavity_worst concave.points_worst symmetry_worst
## 842302            0.7119               0.2654         0.4601
## 842517            0.2416               0.1860         0.2750
## 84300903          0.4504               0.2430         0.3613
## 84348301          0.6869               0.2575         0.6638
## 84358402          0.4000               0.1625         0.2364
## 843786            0.5355               0.1741         0.3985
##          fractal_dimension_worst  X
## 842302                   0.11890 NA
## 842517                   0.08902 NA
## 84300903                 0.08758 NA
## 84348301                 0.17300 NA
## 84358402                 0.07678 NA
## 843786                   0.12440 NA
```

```
#Remove diagnosis column as that is essentially the "answer" our unsupervised
#learning will be looking for, preserve diagnosis as a factor vector for later
wisc.data <- wisc.df[,-1]
#Remove NA column
wisc.data <- wisc.data[,-length(colnames(wisc.data))]
head(wisc.data)
```

```
##          radius_mean texture_mean perimeter_mean area_mean smoothness_mean
## 842302         17.99        10.38         122.80    1001.0         0.11840
## 842517         20.57        17.77         132.90    1326.0         0.08474
## 84300903       19.69        21.25         130.00    1203.0         0.10960
## 84348301       11.42        20.38          77.58     386.1         0.14250
## 84358402       20.29        14.34         135.10    1297.0         0.10030
## 843786         12.45        15.70          82.57     477.1         0.12780
##          compactness_mean concavity_mean concave.points_mean symmetry_mean
## 842302            0.27760         0.3001             0.14710        0.2419
## 842517            0.07864         0.0869             0.07017        0.1812
## 84300903          0.15990         0.1974             0.12790        0.2069
## 84348301          0.28390         0.2414             0.10520        0.2597
## 84358402          0.13280         0.1980             0.10430        0.1809
## 843786            0.17000         0.1578             0.08089        0.2087
##          fractal_dimension_mean radius_se texture_se perimeter_se area_se
## 842302                  0.07871    1.0950     0.9053        8.589  153.40
## 842517                  0.05667    0.5435     0.7339        3.398   74.08
## 84300903                0.05999    0.7456     0.7869        4.585   94.03
## 84348301                0.09744    0.4956     1.1560        3.445   27.23
## 84358402                0.05883    0.7572     0.7813        5.438   94.44
## 843786                  0.07613    0.3345     0.8902        2.217   27.19
##          smoothness_se compactness_se concavity_se concave.points_se
## 842302        0.006399        0.04904      0.05373           0.01587
## 842517        0.005225        0.01308      0.01860           0.01340
## 84300903      0.006150        0.04006      0.03832           0.02058
## 84348301      0.009110        0.07458      0.05661           0.01867
## 84358402      0.011490        0.02461      0.05688           0.01885
## 843786        0.007510        0.03345      0.03672           0.01137
##          symmetry_se fractal_dimension_se radius_worst texture_worst
## 842302       0.03003             0.006193        25.38         17.33
## 842517       0.01389             0.003532        24.99         23.41
## 84300903     0.02250             0.004571        23.57         25.53
## 84348301     0.05963             0.009208        14.91         26.50
## 84358402     0.01756             0.005115        22.54         16.67
## 843786       0.02165             0.005082        15.47         23.75
##          perimeter_worst area_worst smoothness_worst compactness_worst
## 842302            184.60     2019.0           0.1622            0.6656
## 842517            158.80     1956.0           0.1238            0.1866
## 84300903          152.50     1709.0           0.1444            0.4245
## 84348301           98.87      567.7           0.2098            0.8663
## 84358402          152.20     1575.0           0.1374            0.2050
## 843786            103.40      741.6           0.1791            0.5249
##          concavity_worst concave.points_worst symmetry_worst
## 842302            0.7119               0.2654         0.4601
## 842517            0.2416               0.1860         0.2750
## 84300903          0.4504               0.2430         0.3613
## 84348301          0.6869               0.2575         0.6638
## 84358402          0.4000               0.1625         0.2364
## 843786            0.5355               0.1741         0.3985
##          fractal_dimension_worst
## 842302                   0.11890
## 842517                   0.08902
## 84300903                 0.08758
## 84348301                 0.17300
## 84358402                 0.07678
## 843786                   0.12440
```

```
diagnosis <- as.vector(wisc.df$diagnosis)
diagnosis_factor <- factor(diagnosis)
```

Q1. How many observations are in this dataset?

```
#The number of observations is equal to the number of rows in the dataset
nrow(wisc.df)
```

```
## [1] 569
```

Q2. How many of the observations have a malignant diagnosis?

```
#Extract number of malignant samples using table
table(diagnosis_factor)["M"]
```

```
##   M
## 212
```

Q3. How many variables/features in the data are suffixed with `_mean`?

```
#Pull the columns that contain "_mean" and count them
mean_columns <- grep("_mean", colnames(wisc.data))
length(mean_columns)
```

```
## [1] 10
```

Principal Component Analysis

```
#Does the data need to be scaled? Check column means and standard deviations
colMeans(wisc.data)
```

```
##             radius_mean            texture_mean          perimeter_mean
##            1.412729e+01            1.928965e+01            9.196903e+01
##               area_mean         smoothness_mean        compactness_mean
##            6.548891e+02            9.636028e-02            1.043410e-01
##          concavity_mean     concave.points_mean           symmetry_mean
##            8.879932e-02            4.891915e-02            1.811619e-01
##  fractal_dimension_mean               radius_se              texture_se
##            6.279761e-02            4.051721e-01            1.216853e+00
##            perimeter_se                 area_se           smoothness_se
##            2.866059e+00            4.033708e+01            7.040979e-03
##          compactness_se            concavity_se       concave.points_se
##            2.547814e-02            3.189372e-02            1.179614e-02
##             symmetry_se    fractal_dimension_se            radius_worst
##            2.054230e-02            3.794904e-03            1.626919e+01
##           texture_worst         perimeter_worst              area_worst
##            2.567722e+01            1.072612e+02            8.805831e+02
##        smoothness_worst       compactness_worst         concavity_worst
##            1.323686e-01            2.542650e-01            2.721885e-01
##    concave.points_worst          symmetry_worst fractal_dimension_worst
##            1.146062e-01            2.900756e-01            8.394582e-02
```

```
apply(wisc.data, 2, sd)
```

```
##             radius_mean            texture_mean          perimeter_mean
##            3.524049e+00            4.301036e+00            2.429898e+01
##               area_mean         smoothness_mean        compactness_mean
##            3.519141e+02            1.406413e-02            5.281276e-02
##          concavity_mean     concave.points_mean           symmetry_mean
##            7.971981e-02            3.880284e-02            2.741428e-02
##  fractal_dimension_mean               radius_se              texture_se
##            7.060363e-03            2.773127e-01            5.516484e-01
##            perimeter_se                 area_se           smoothness_se
##            2.021855e+00            4.549101e+01            3.002518e-03
##          compactness_se            concavity_se       concave.points_se
##            1.790818e-02            3.018606e-02            6.170285e-03
##             symmetry_se    fractal_dimension_se            radius_worst
##            8.266372e-03            2.646071e-03            4.833242e+00
##           texture_worst         perimeter_worst              area_worst
##            6.146258e+00            3.360254e+01            5.693570e+02
##        smoothness_worst       compactness_worst         concavity_worst
##            2.283243e-02            1.573365e-01            2.086243e-01
##    concave.points_worst          symmetry_worst fractal_dimension_worst
##            6.573234e-02            6.186747e-02            1.806127e-02
```

The data has a wide range of standard deviations, so it should be scaled on a per column basis so the ones with higher variance don't automatically contribute more to the PCA.

```r
#Run PCA and look at summary
wisc.pr <- prcomp(wisc.data, scale=TRUE)
summary(wisc.pr)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     3.6444 2.3857 1.67867 1.40735 1.28403 1.09880 0.82172
## Proportion of Variance 0.4427 0.1897 0.09393 0.06602 0.05496 0.04025 0.02251
## Cumulative Proportion  0.4427 0.6324 0.72636 0.79239 0.84734 0.88759 0.91010
##                            PC8    PC9    PC10   PC11    PC12    PC13    PC14
## Standard deviation     0.69037 0.6457 0.59219 0.5421 0.51104 0.49128 0.39624
## Proportion of Variance 0.01589 0.0139 0.01169 0.0098 0.00871 0.00805 0.00523
## Cumulative Proportion  0.92598 0.9399 0.95157 0.9614 0.97007 0.97812 0.98335
##                           PC15    PC16    PC17    PC18    PC19    PC20   PC21
## Standard deviation     0.30681 0.28260 0.24372 0.22939 0.22244 0.17652 0.1731
## Proportion of Variance 0.00314 0.00266 0.00198 0.00175 0.00165 0.00104 0.0010
## Cumulative Proportion  0.98649 0.98915 0.99113 0.99288 0.99453 0.99557 0.9966
##                           PC22    PC23   PC24    PC25    PC26    PC27    PC28
## Standard deviation     0.16565 0.15602 0.1344 0.12442 0.09043 0.08307 0.03987
## Proportion of Variance 0.00091 0.00081 0.0006 0.00052 0.00027 0.00023 0.00005
## Cumulative Proportion  0.99749 0.99830 0.9989 0.99942 0.99969 0.99992 0.99997
##                           PC29    PC30
## Standard deviation     0.02736 0.01153
## Proportion of Variance 0.00002 0.00000
## Cumulative Proportion  1.00000 1.00000
```

> Q4. From your results, what proportion of the original variance is captured by the first principal components (PC1)?

As seen in the summary above, ~44% of the original variance is captured by PC1.

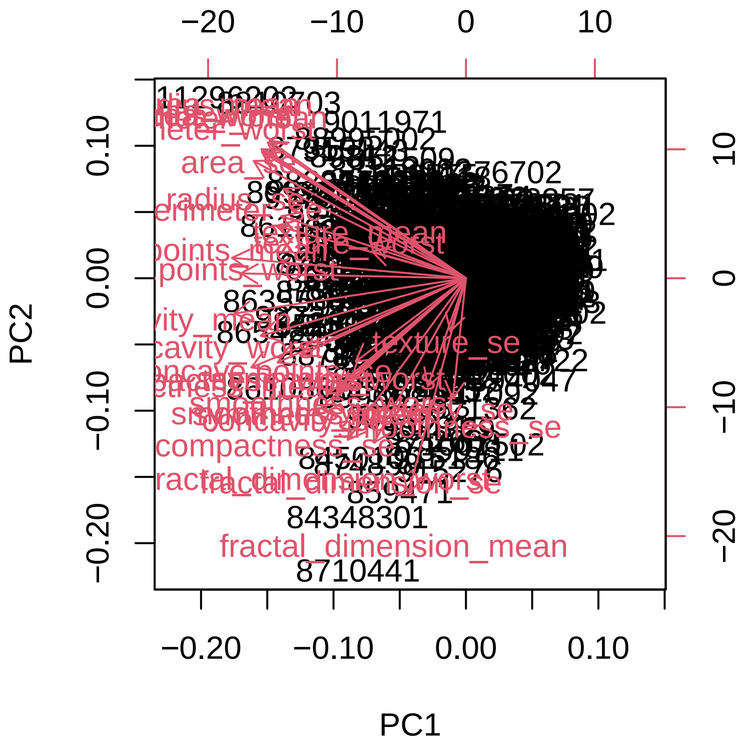
> Q5. How many principal components (PCs) are required to describe at least 70% of the original variance in the data?

At least 3 PCs are required to describe 70% of the variance (see cumulative proportion in summary).

> Q6. How many principal components (PCs) are required to describe at least 90% of the original variance in the data?

At least 7 PCs are required to describe 90% of the variance (see cumulative proportion in summary).

```r
#Let's plot the PCA!
biplot(wisc.pr)
```

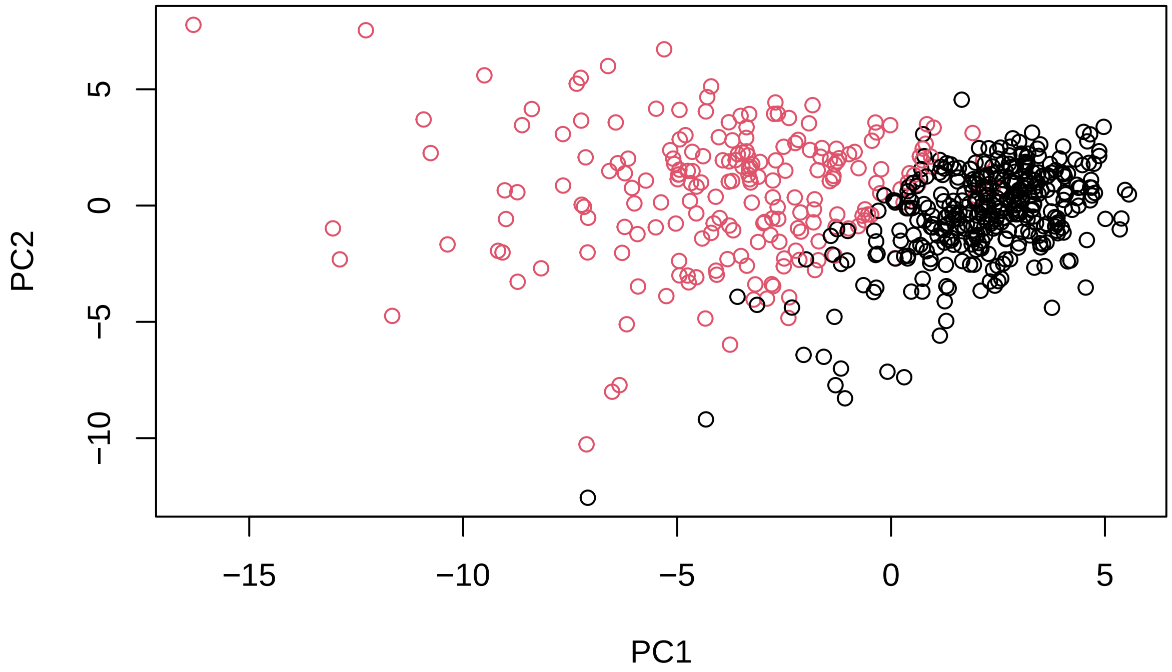


Q7. What stands out to you about this plot? Is it easy or difficult to understand? Why?

This plot is impossible to read and cannot show us much of anything.

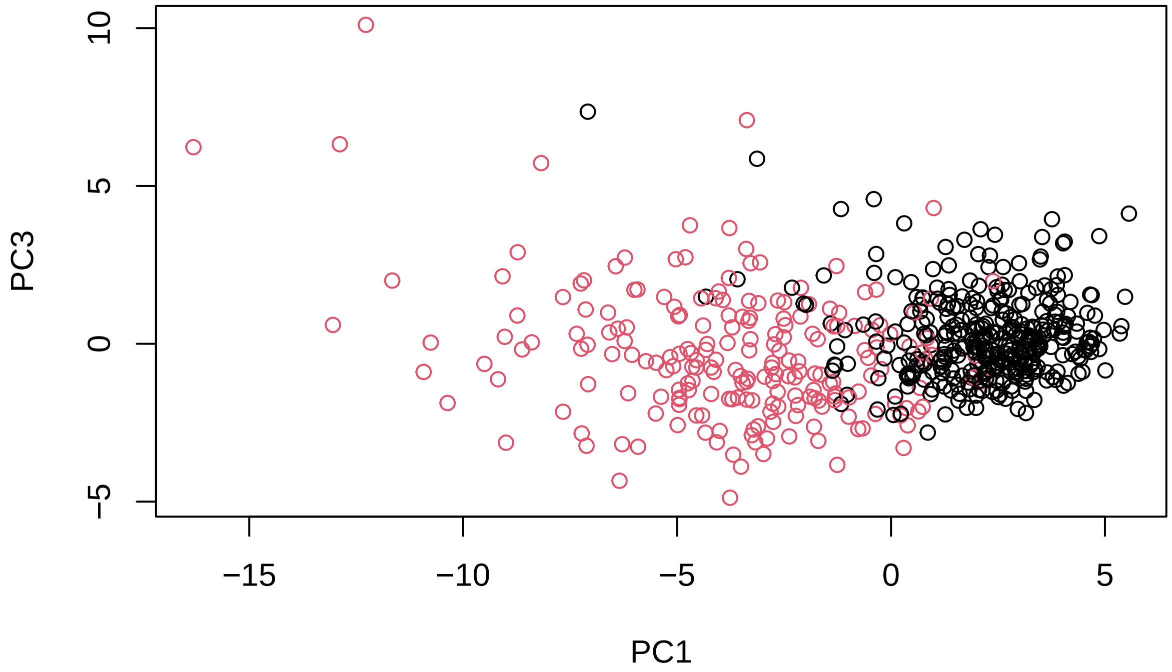
Instead, let's plot a scatter plot colored by diagnosis.

```
plot(wisc.pr$x[,1:2], col = diagnosis_factor,
     xlab = "PC1", ylab = "PC2")
```



Q8. Generate a similar plot for principal components 1 and 3. What do you notice about these plots?

```
# Repeat for components 1 and 3
plot(wisc.pr$x[,c(1,3)], col = diagnosis_factor,
     xlab = "PC1", ylab = "PC3")
```



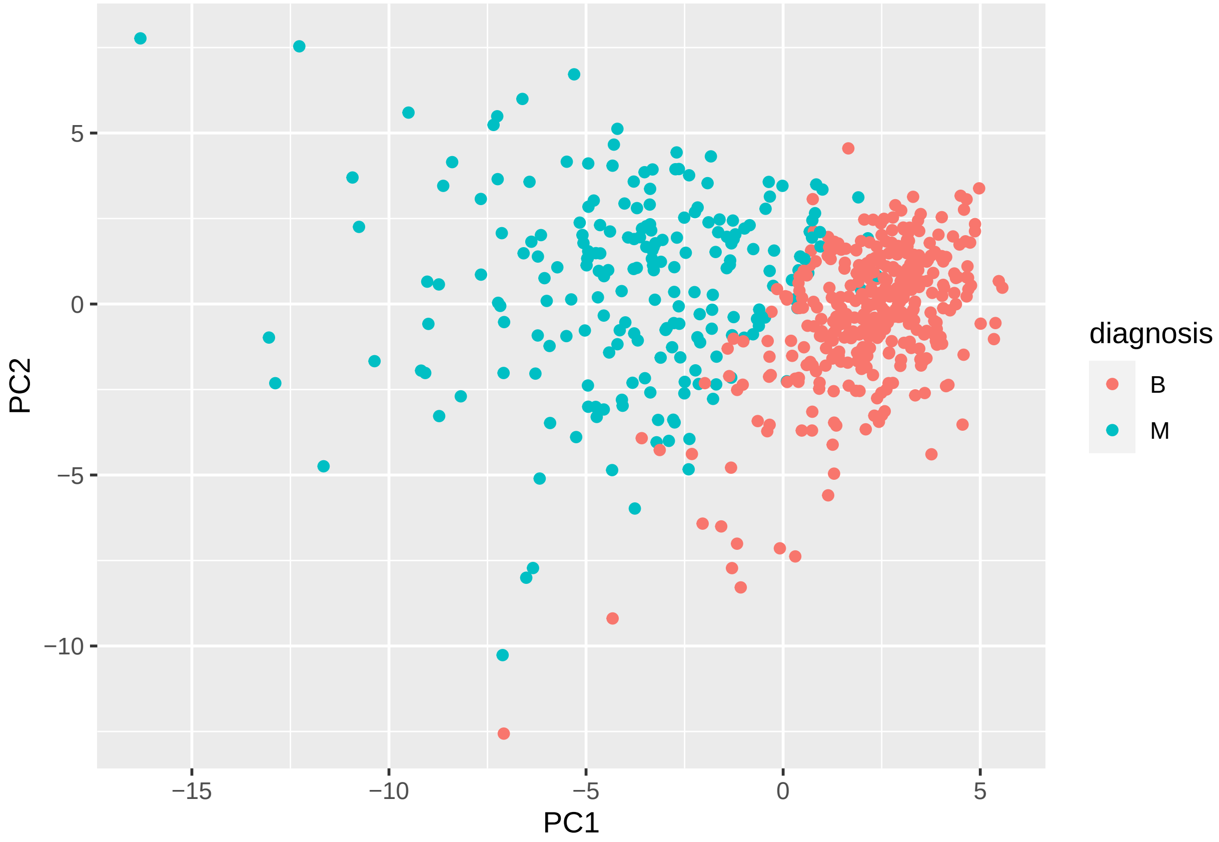
This plot has much poorer separation of the two clusters than the first, because PC3 captures less variance than PC2.

Making Nicer Plots With Ggplot

```
# Create a data.frame for ggplot
wisc.pr.df <- as.data.frame(wisc.pr$x)
#wisc.pr.df$diagnosis <- diagnosis_factor

# Load the ggplot2 package
library(ggplot2)

# Make a scatter plot colored by diagnosis
ggplot(wisc.pr.df) +
  aes(PC1, PC2, col=diagnosis) +
  geom_point()
```



What is the variance in each PC?

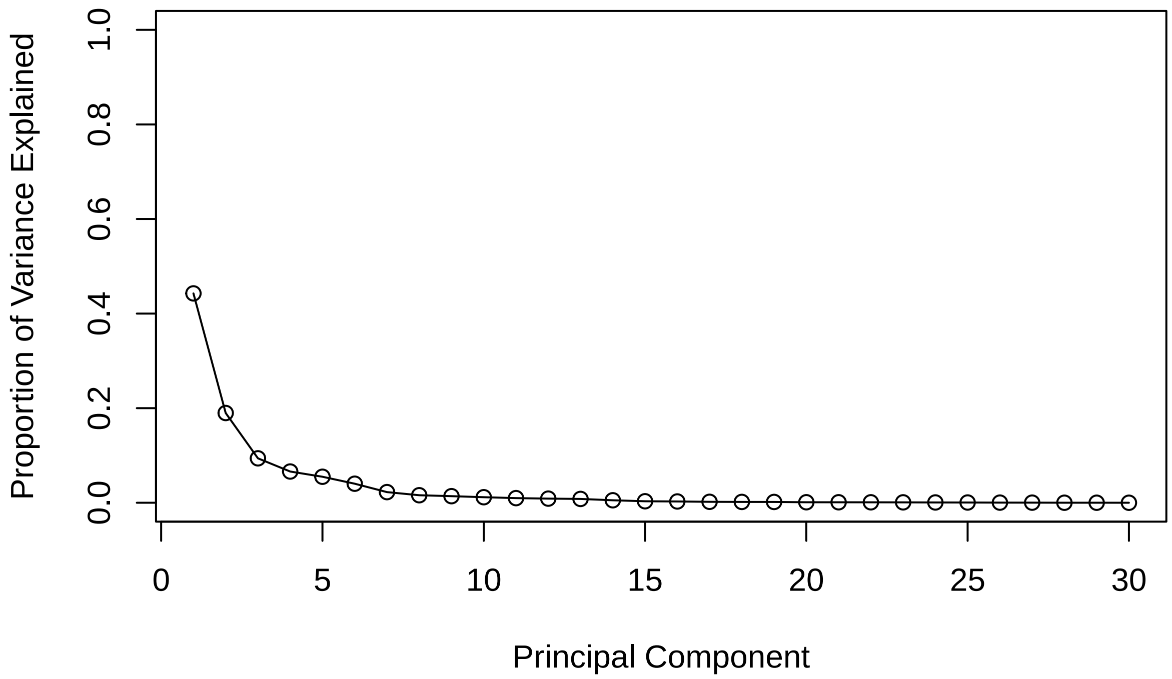
```
# Calculate variance of each component
pr.var <- wisc.pr$sdev^2
pr.var
```

```
##  [1] 1.328161e+01 5.691355e+00 2.817949e+00 1.980640e+00 1.648731e+00
##  [6] 1.207357e+00 6.752201e-01 4.766171e-01 4.168948e-01 3.506935e-01
## [11] 2.939157e-01 2.611614e-01 2.413575e-01 1.570097e-01 9.413497e-02
## [16] 7.986280e-02 5.939904e-02 5.261878e-02 4.947759e-02 3.115940e-02
## [21] 2.997289e-02 2.743940e-02 2.434084e-02 1.805501e-02 1.548127e-02
## [26] 8.177640e-03 6.900464e-03 1.589338e-03 7.488031e-04 1.330448e-04
```

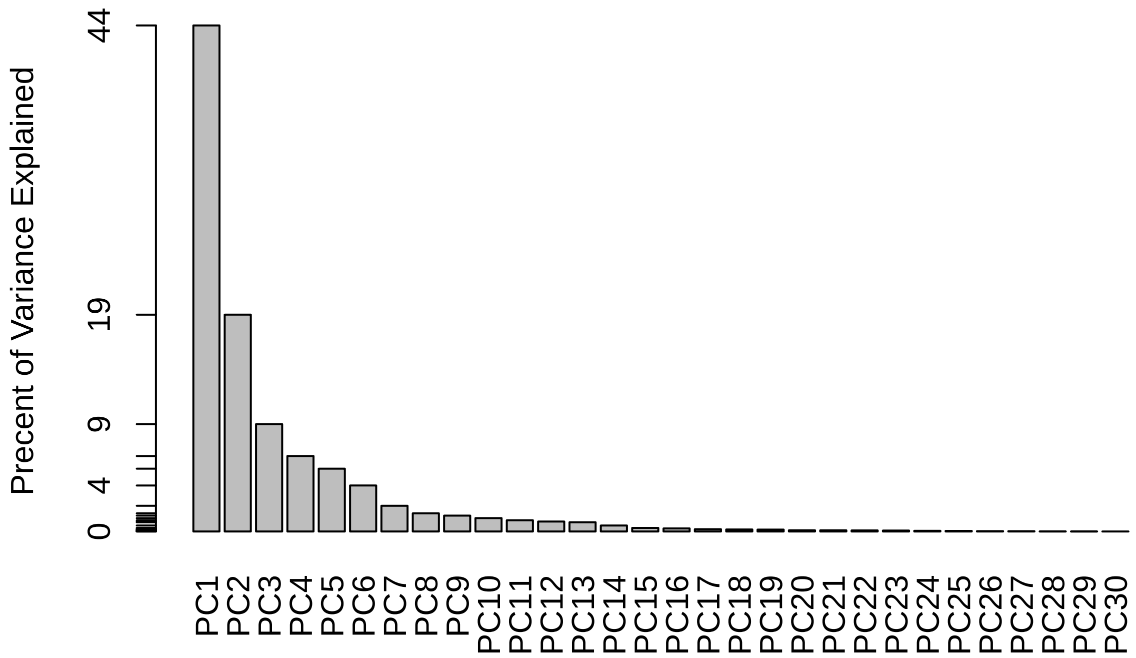
What is the proportion of variance explained by each PC? Let’s look at some plots of this.

```
# Variance explained by each principal component: pve
pve <- pr.var/sum(pr.var)

# Plot variance explained for each principal component
plot(pve, xlab = "Principal Component",
     ylab = "Proportion of Variance Explained",
     ylim = c(0, 1), type = "o")
```



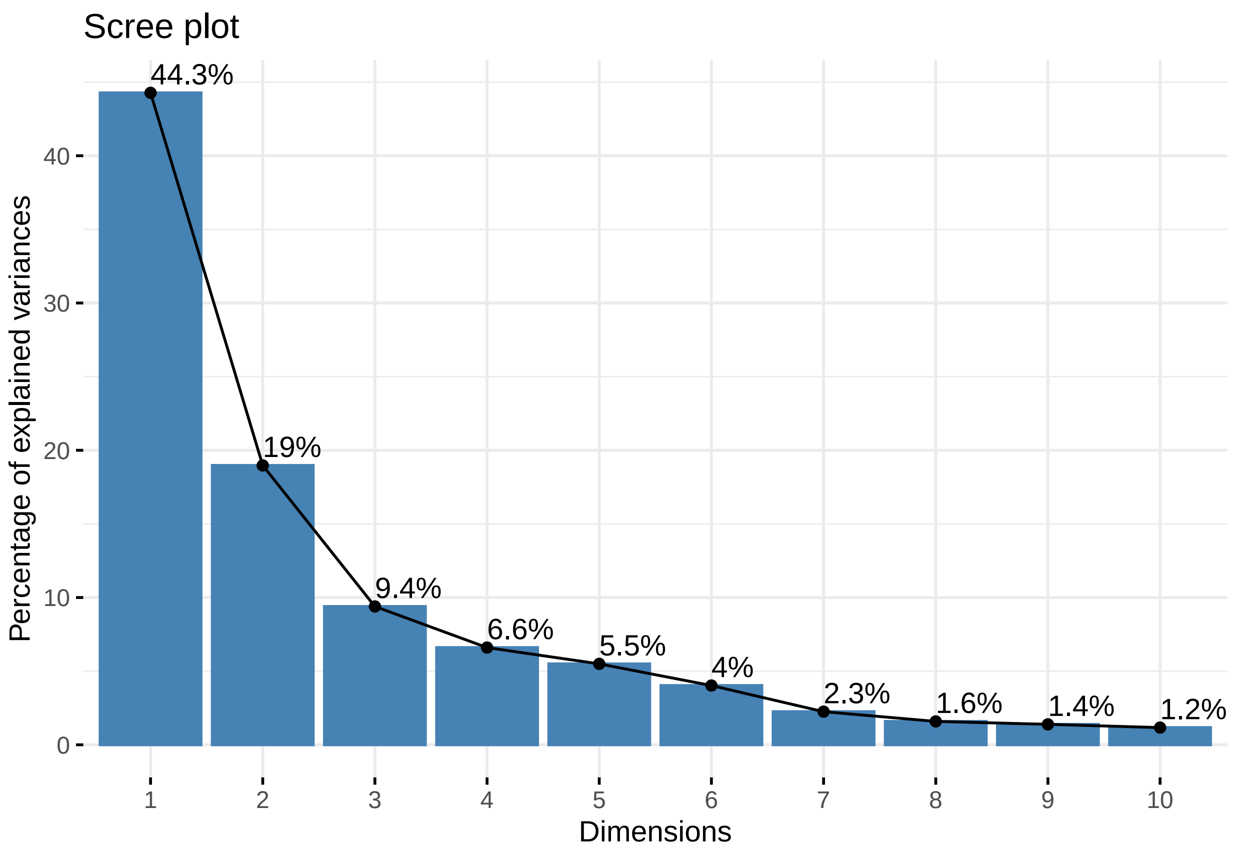
```
# Alternative scree plot of the same data, note data driven y-axis
barplot(pve, ylab = "Precent of Variance Explained",
     names.arg=paste0("PC",1:length(pve)), las=2, axes = FALSE)
axis(2, at=pve, labels=round(pve,2)*100)
```



```
#ggplot based graph
library(factoextra)
```

```
## Welcome! Want to learn more? See two factoextra-related books at https://goo.gl/ve3WBa
```

```
fviz_eig(wisc.pr, addlabels = TRUE)
```



Q9. For the first principal component, what is the component of the loading vector (i.e. wisc.pr$rotation[,1]) for the feature concave.points_mean?

```
wisc.pr$rotation[,1]["concave.points_mean"]
```

```
## concave.points_mean 
##          -0.2608538
```

```
#This number represents how much "concave.points_mean" contributes to PC1
```

Q10. What is the minimum number of principal components required to explain 80% of the variance of the data?

```
#Sum each PC's variance until 80% is reached
sum <- 0
for (i in 1:length(pve)){
  sum <- sum + pve[i]
  if (sum >= 0.80){
    print(i)
    break
  }
}
```

```
## [1] 5
```

```
#Note this can be determined more easily by looking at the summary(wisc.pr)
#results
```

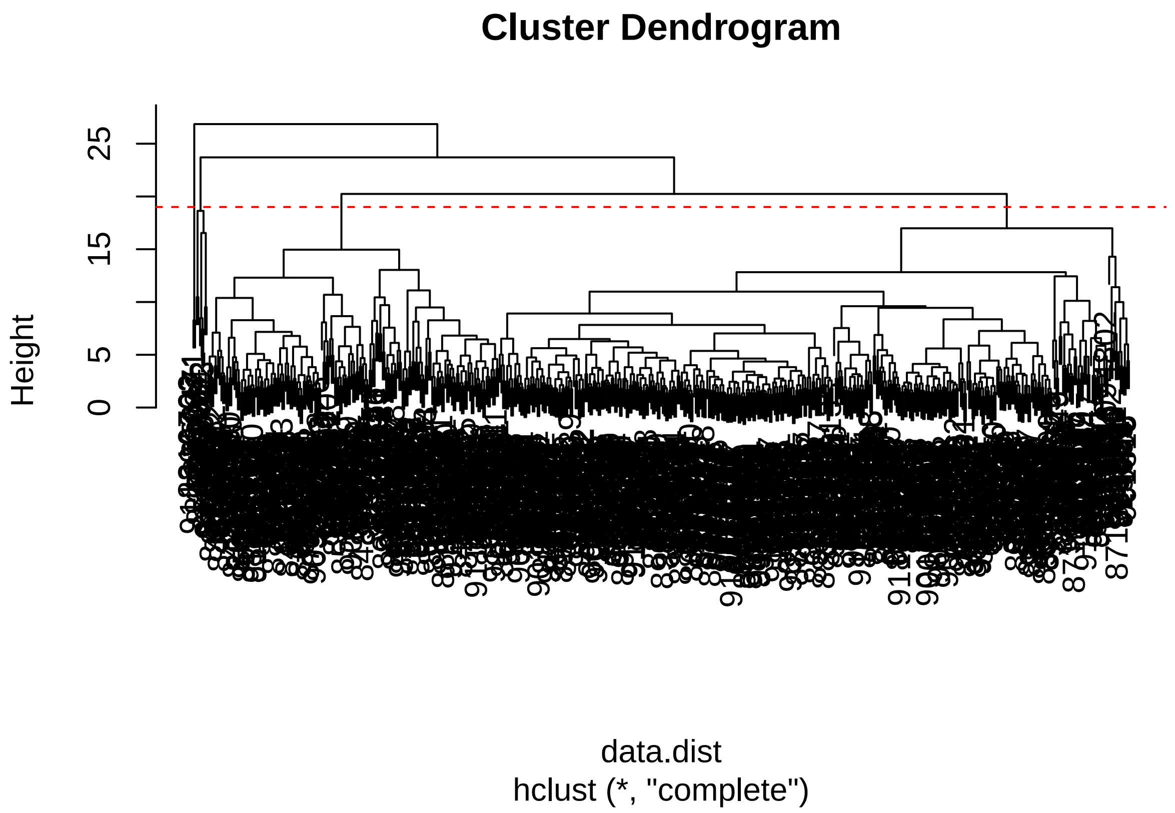
Hierarchical Clustering

```
# Scale the wisc.data data using the "scale()" function
data.scaled <- scale(wisc.data)
#Calculate distance (euclidean)
data.dist <- dist(data.scaled)
#Do the clustering
wisc.hclust <- hclust(data.dist, method="complete")
```

> Q11. Using the plot() and abline() functions, what is the height at which the clustering model has 4 clusters?

Time to plot the cluster dendrogram.

```
plot(wisc.hclust)
abline(h=19, col="red", lty=2)
```



A line height between 19 and 20 cuts the data into 4 clusters.

```
#We can also set the number of clusters
wisc.hclust.clusters <- cutree(wisc.hclust, 4)
#Compare clustes to diagnosis
table(wisc.hclust.clusters, diagnosis)
```

```
##                       diagnosis
## wisc.hclust.clusters   B   M
##                    1  12 165
##                    2   2   5
##                    3 343  40
##                    4   0   2
```

> Q12. Can you find a better cluster vs diagnoses match by cutting into a different number of clusters between 2 and 10?

```
#Let's try cutting into fewer clusters and see how well it matches diagnosis
#Three?
wisc.hclust.clusters <- cutree(wisc.hclust, 3)
table(wisc.hclust.clusters, diagnosis)
```

```
##                       diagnosis
## wisc.hclust.clusters   B   M
##                    1 355 205
##                    2   2   5
##                    3   0   2
```

```
#Two?
wisc.hclust.clusters <- cutree(wisc.hclust, 2)
table(wisc.hclust.clusters, diagnosis)
```

```
##                       diagnosis
## wisc.hclust.clusters   B   M
##                    1 357 210
##                    2   0   2
```

```
#Both of these fail, with benign and malignant largely clustering together
#Let's try more
#Five?
wisc.hclust.clusters <- cutree(wisc.hclust, 5)
table(wisc.hclust.clusters, diagnosis)
```

```
##                       diagnosis
## wisc.hclust.clusters   B   M
##                    1  12 165
##                    2   0   5
##                    3 343  40
##                    4   2   0
##                    5   0   2
```

```r
#Ten?
wisc.hclust.clusters <- cutree(wisc.hclust, 10)
table(wisc.hclust.clusters, diagnosis)
```

```
##                      diagnosis
## wisc.hclust.clusters   B   M
##                   1   12  86
##                   2    0  59
##                   3    0   3
##                   4  331  39
##                   5    0  20
##                   6    2   0
##                   7   12   0
##                   8    0   2
##                   9    0   2
##                   10   0   1
```

```r
#Even ten clusters can't separate fully into exclusively malignant and benign
#clusters, four clusters seems best
```

Q13. Which method gives your favorite results for the same data.dist dataset? Explain your reasoning.

```r
#Let's construct hclust objects using the various methods
wisc.hclust.complete <- hclust(data.dist, method="complete")
wisc.hclust.single <- hclust(data.dist, method="single")
wisc.hclust.average <- hclust(data.dist, method="average")
wisc.hclust.ward <- hclust(data.dist, method="ward.D2")

#Can any of the four give two good clusters?
wisc.hclust.complete.clusters_2 <- cutree(wisc.hclust.complete, 2)
table(wisc.hclust.complete.clusters_2, diagnosis)
```

```
##                                 diagnosis
## wisc.hclust.complete.clusters_2   B   M
##                               1 357 210
##                               2   0   2
```

```r
wisc.hclust.single.clusters_2 <- cutree(wisc.hclust.single, 2)
table(wisc.hclust.single.clusters_2, diagnosis)
```

```
##                               diagnosis
## wisc.hclust.single.clusters_2   B   M
##                             1 357 210
##                             2   0   2
```

```r
wisc.hclust.average.clusters_2 <- cutree(wisc.hclust.average, 2)
table(wisc.hclust.average.clusters_2, diagnosis)
```

```
##                                diagnosis
## wisc.hclust.average.clusters_2   B   M
##                              1 357 209
##                              2   0   3
```

```r
wisc.hclust.ward.clusters_2 <- cutree(wisc.hclust.ward, 2)
table(wisc.hclust.ward.clusters_2, diagnosis)
```

```
##                            diagnosis
## wisc.hclust.ward.clusters_2   B   M
##                           1  20 164
##                           2 337  48
```

```r
#Ward clusters fairly well already with just two! How does each do with 4?
wisc.hclust.complete.clusters_4 <- cutree(wisc.hclust.complete, 4)
table(wisc.hclust.complete.clusters_4, diagnosis)
```

```
##                                diagnosis
## wisc.hclust.complete.clusters_4   B   M
##                               1  12 165
##                               2   2   5
##                               3 343  40
##                               4   0   2
```

```r
wisc.hclust.single.clusters_4 <- cutree(wisc.hclust.single, 4)
table(wisc.hclust.single.clusters_4, diagnosis)
```

```
##                              diagnosis
## wisc.hclust.single.clusters_4   B   M
##                             1 356 209
##                             2   1   0
##                             3   0   2
##                             4   0   1
```

```r
wisc.hclust.average.clusters_4 <- cutree(wisc.hclust.average, 4)
table(wisc.hclust.average.clusters_4, diagnosis)
```

```
##                               diagnosis
## wisc.hclust.average.clusters_4   B   M
##                              1 355 209
##                              2   2   0
##                              3   0   1
##                              4   0   2
```

```r
wisc.hclust.ward.clusters_4 <- cutree(wisc.hclust.ward, 4)
table(wisc.hclust.ward.clusters_4, diagnosis)
```

```
##                            diagnosis
## wisc.hclust.ward.clusters_4   B   M
##                           1   0 115
##                           2   6  48
##                           3 337  48
##                           4  14   1
```

```
#Ward works about as well as complete does with 4 clusters, the other two are
#nowhere near as good
```

The method "ward.D2" gives me my favorite results because it is the only one able to produce decent clusters when looking for 2 clusters, and the only one able to produce a pure cluster when looking for 4 clusters (cluster 1, all malignant).

K-means Clustering

```
#Run K-means on the data with 2 centers, 20 times
wisc.km <- kmeans(data.scaled, centers= 2, nstart= 20)
table(wisc.km$cluster, diagnosis_factor)
```

```
##    diagnosis_factor
##       B   M
##   1  14 175
##   2 343  37
```

> Q14. How well does k-means separate the two diagnoses? How does it compare to your hclust results?

The K-means method clusters the data slightly better (less "misdiagnosed" malignant samples in the benign cluster and vice versa as compared with the "ward.D2" method)

```
#How do the clusters relate?
table(wisc.hclust.ward.clusters_2, wisc.km$cluster)
```

```
##
## wisc.hclust.ward.clusters_2   1   2
##                           1 173  11
##                           2  16 369
```

```
table(wisc.hclust.ward.clusters_4, wisc.km$cluster)
```

```
##
## wisc.hclust.ward.clusters_4   1   2
##                           1 113   2
##                           2  50   4
##                           3  16 369
##                           4  10   5
```

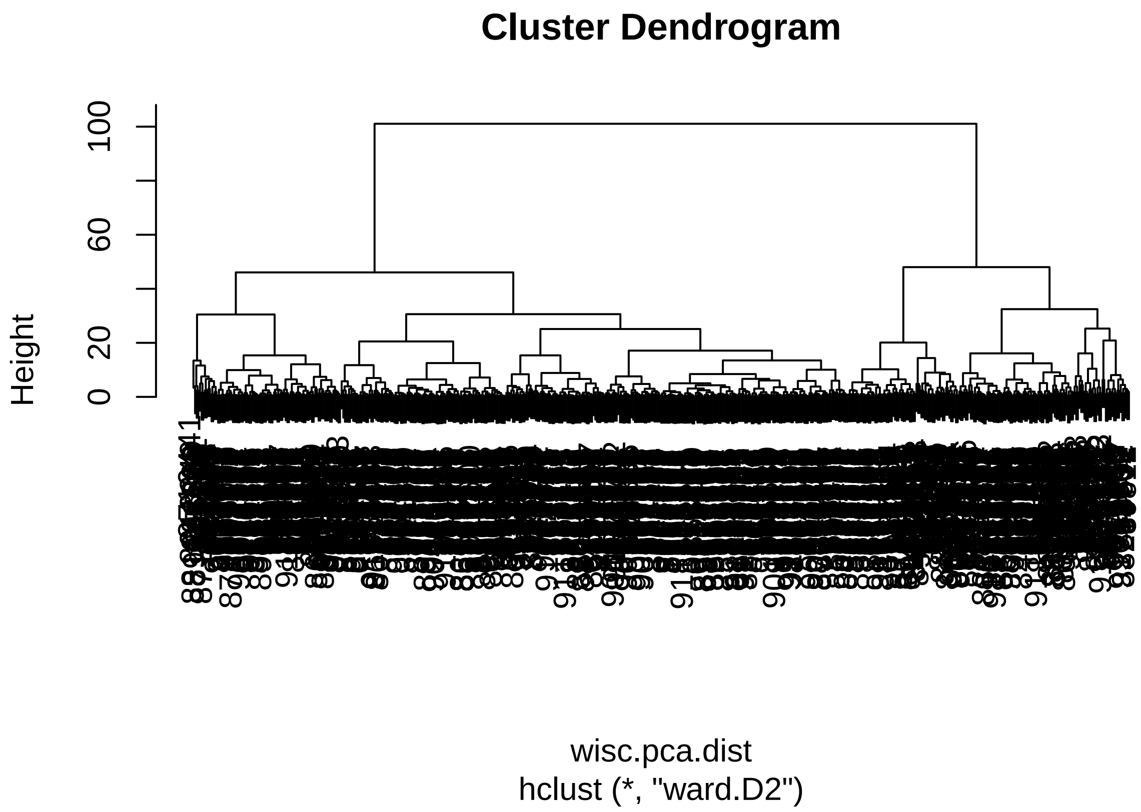
```
table(wisc.hclust.complete.clusters_4, wisc.km$cluster)
```

```
##
## wisc.hclust.complete.clusters_4   1   2
##                               1 160  17
##                               2   7   0
##                               3  20 363
##                               4   2   0
```

```
#Each hclust clustering correlates fairly well with the k-means clustering
```

Combining Methods: Clustering with PCA results

```
#Run hclust on the PCA results with 4 PCs
wisc.pca.dist <- dist(wisc.pr$x[,1:4])
wisc.pr.hclust <- hclust(wisc.pca.dist, method="ward.D2")
plot(wisc.pr.hclust)
```



```
#Group the results into 2 clusters
grps <- cutree(wisc.pr.hclust, k=2)
table(grps)
```

```
## grps
##   1   2
## 171 398
```

```
#Compare grps to diagnosis
table(grps, diagnosis_factor)
```

```
##     diagnosis_factor
## grps   B   M
##    1   6 165
##    2 351  47
```

```
#BONUS: how many PCs should we cluster on to minimize false negatives (min
#number of malignant samples in the benign cluster)

#Initialize variables
best_PCs <- 0
false_neg <- 400

#Test each number of PCs
for (i in 1:30){

  #Cluster as before
  temp.wisc.pca.dist <- dist(wisc.pr$x[,1:i])
  temp.wisc.pr.hclust <- hclust(temp.wisc.pca.dist, method="ward.D2")
  temp.grps <- cutree(temp.wisc.pr.hclust, k=2)

  #Read in the false negatives
  temp.false_neg <- min(table(temp.grps, diagnosis_factor)[1, "M"], table(temp.grps, diagnosis_factor)[2

  #Update if the nub=mber of false negatives is fewer than the current min
  if (temp.false_neg < false_neg){
    false_neg <- temp.false_neg
    best_PCs <- i
  }
}
```

> Q15. How well does the newly created model with two clusters (optimized for 10 PCs) separate out the two diagnoses?

```
#Output the number of PCs
print(best_PCs)
```

```
## [1] 10
```

```
#Produce clusters
wisc.pca.dist <- dist(wisc.pr$x[,1:best_PCs])
wisc.pr.hclust <- hclust(wisc.pca.dist, method="ward.D2")
grps <- cutree(wisc.pr.hclust, k=2)

#Output diagnosis table
table(grps, diagnosis_factor)
```

```
##     diagnosis_factor
## grps   B   M
##    1  39 205
##    2 318   7
```

The new model correlates very strongly with the proper diagnosis and minimizes the number of false negatives.

Q16. How well do the k-means and hierarchical clustering models you created in previous sections (i.e. before PCA) do in terms of separating the diagnoses? Again, use the table() function to compare the output of each model (wisc.km$cluster and wisc.hclust.clusters) with the vector containing the actual diagnoses.

```
table(wisc.km$cluster, diagnosis)
```

```
##    diagnosis
##       B   M
##   1  14 175
##   2 343  37
```

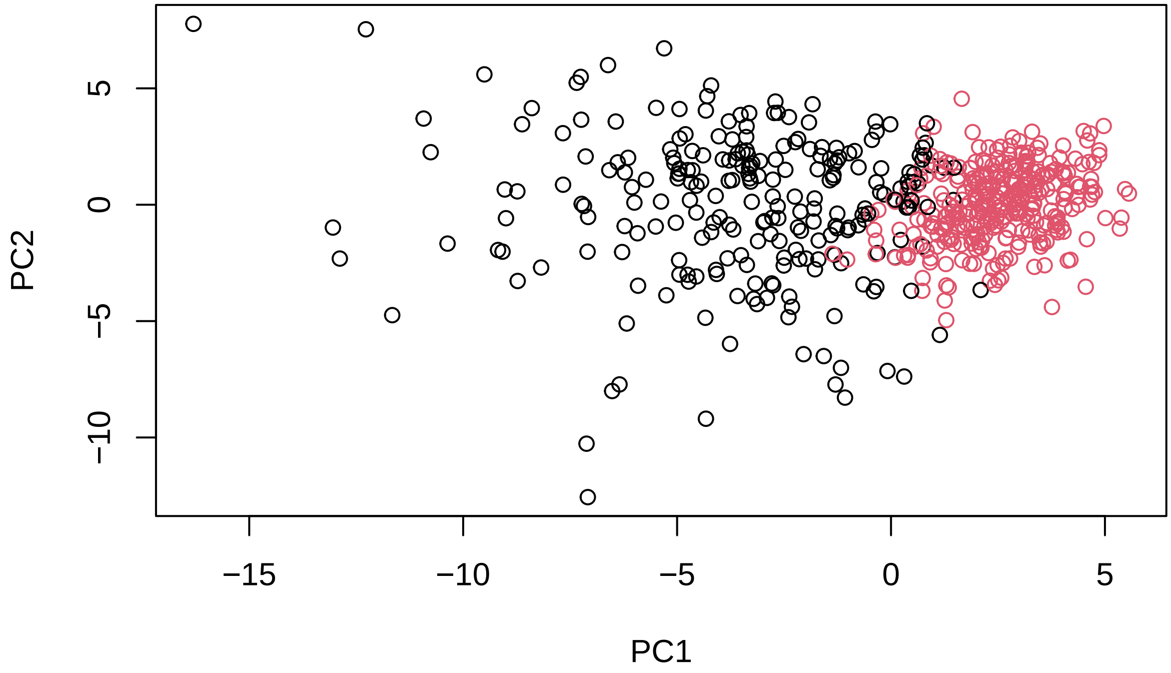
```
table(wisc.hclust.ward.clusters_2, diagnosis)
```

```
##                              diagnosis
## wisc.hclust.ward.clusters_2   B   M
##                           1  20 164
##                           2 337  48
```

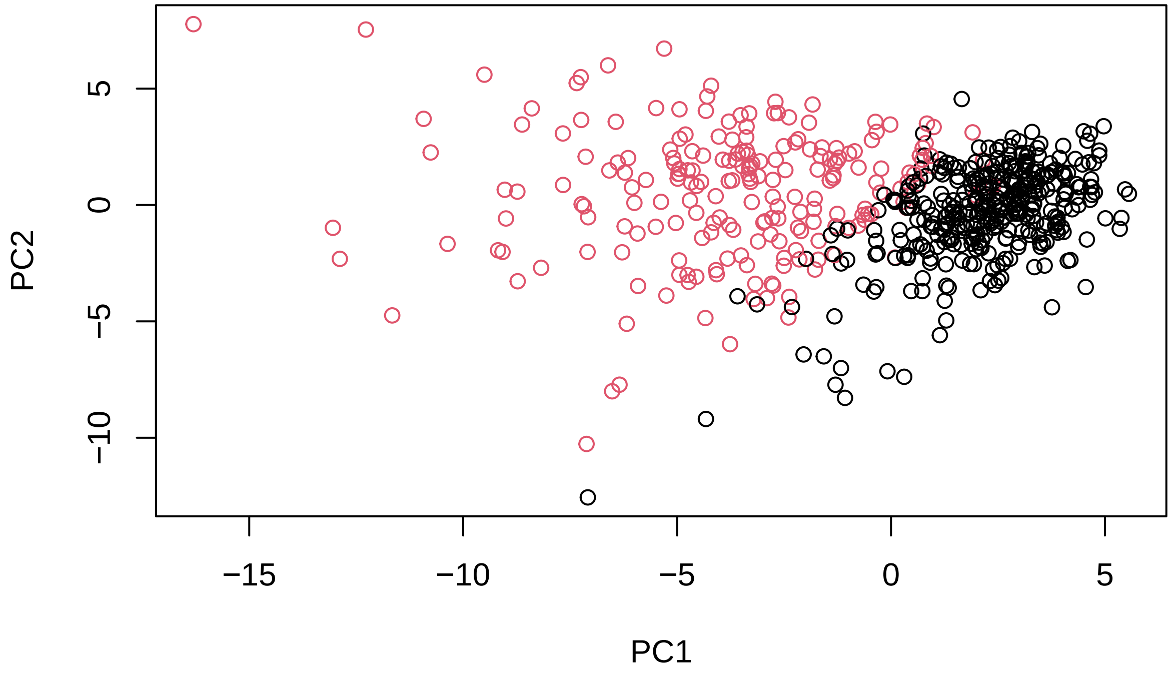
These two methods also correlate well, but the PCA clustering is more accurate and can be more easily fined tuned to minimize false negatives.

For fun, let's plot the samples colored by the "best" grps and then by diagnosis.

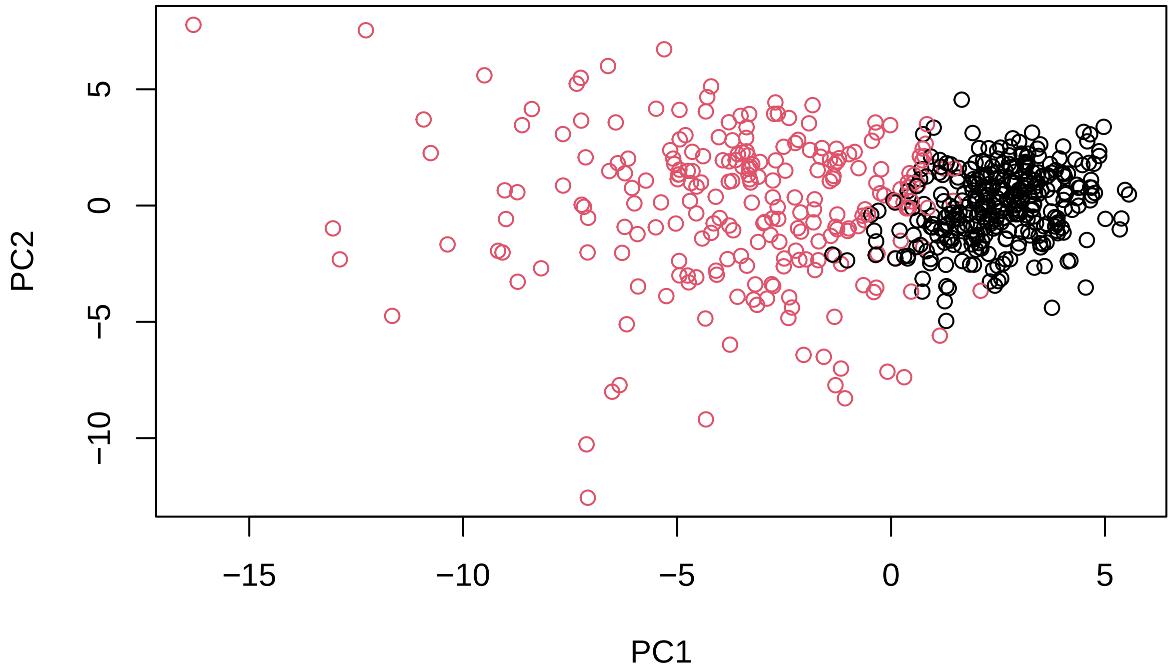
```
plot(wisc.pr$x[,1:2], col=grps)
```



```
plot(wisc.pr$x[,1:2], col=diagnosis_factor)
```



```
#Change colors to match
g <- as.factor(grps)
g <- relevel(g,2)
plot(wisc.pr$x[,1:2], col=g)
```



Q17. Which of your analysis procedures resulted in a clustering model with the best specificity? How about sensitivity?

```
#Calculate sensitivity for each model

#K-means sensitivity
Ksens <- 175/(175+37)

#Hclust sensitivity
Hsens <- 164/(164+48)

#PCA sensitivity
Psens <- 205/(205+7)

#What's best?
sens_vec <-c(Ksens=Ksens, Hsens=Hsens, Psens=Psens)
sens_vec
```

```
##     Ksens     Hsens     Psens
## 0.8254717 0.7735849 0.9669811
```

```
which.max(sens_vec)
```

```
## Psens
##     3
```

```
max(sens_vec)
```

```
## [1] 0.9669811
```

```
#Calculate specificity for each model

#K-means specificity
Kspec <- 343/(343+14)

#Hclust specificity
Hspec <- 337/(337+20)

#PCA specificity
Pspec <- 318/(318+39)

#What's best?
spec_vec <-c(Kspec=Kspec, Hspec=Hspec, Pspec=Pspec)
spec_vec
```

```
##     Kspec     Hspec     Pspec
## 0.9607843 0.9439776 0.8907563
```

```
which.max(spec_vec)
```

```
## Kspec
##     1
```

```
max(spec_vec)
```

```
## [1] 0.9607843
```

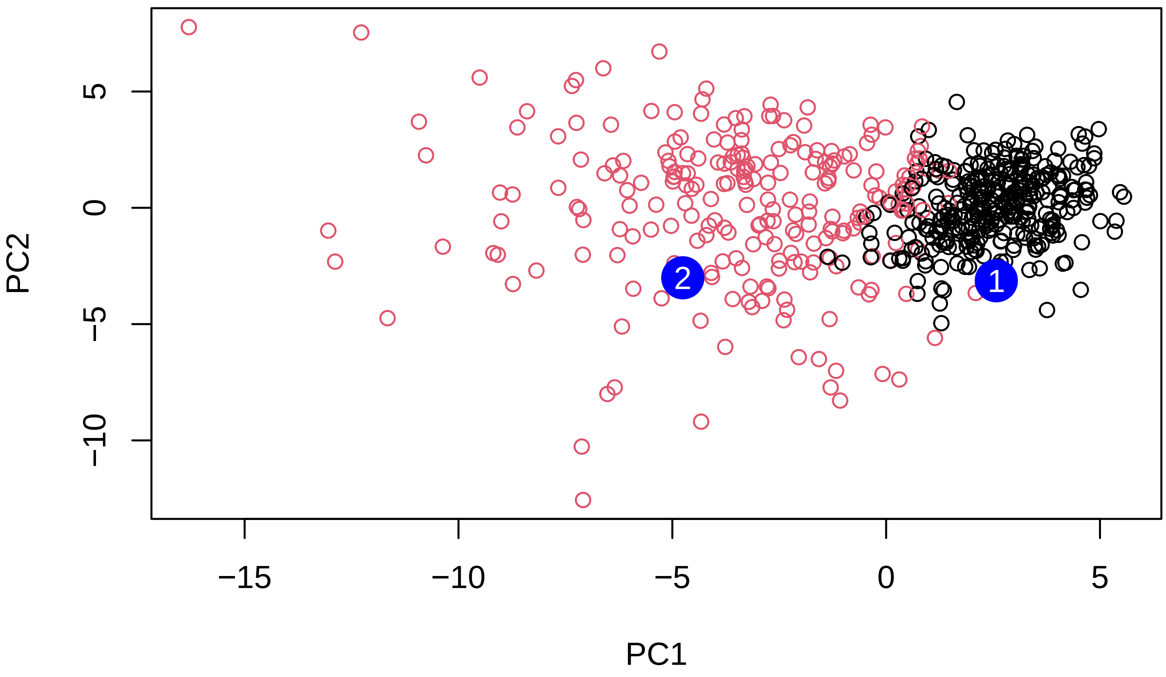
The PCA model is the most sensitive, but sacrifices specificity (k-means is best in terms of specificity, but I still believe the PCA is best for diagnostic purposes).

Can we now predict diagnoses on new data?

```
#Read in new samples
url <- "https://tinyurl.com/new-samples-CSV"
new <- read.csv(url)
npc <- predict(wisc.pr, newdata=new)
npc
```

```
##            PC1       PC2        PC3        PC4       PC5        PC6        PC7
## [1,]  2.576616 -3.135913  1.3990492 -0.7631950  2.781648 -0.8150185 -0.3959098
## [2,] -4.754928 -3.009033 -0.1660946 -0.6052952 -1.140698 -1.2189945  0.8193031
##             PC8       PC9       PC10      PC11      PC12      PC13     PC14
## [1,] -0.2307350 0.1029569 -0.9272861 0.3411457  0.375921 0.1610764 1.187882
## [2,] -0.3307423 0.5281896 -0.4855301 0.7173233 -1.185917 0.5893856 0.303029
##            PC15       PC16        PC17        PC18        PC19       PC20
## [1,] 0.3216974 -0.1743616 -0.07875393 -0.11207028 -0.08802955 -0.2495216
## [2,] 0.1299153  0.1448061 -0.40509706  0.06565549  0.25591230 -0.4289500
##             PC21       PC22       PC23       PC24        PC25         PC26
## [1,]  0.1228233 0.09358453 0.08347651  0.1223396  0.02124121  0.078884581
## [2,] -0.1224776 0.01732146 0.06316631 -0.2338618 -0.20755948 -0.009833238
##              PC27        PC28         PC29         PC30
## [1,]  0.220199544 -0.02946023 -0.015620933  0.005269029
## [2,] -0.001134152  0.09638361  0.002795349 -0.019015820
```

```
#Plot the new samples
plot(wisc.pr$x[,1:2], col=g)
points(npc[,1], npc[,2], col="blue", pch=16, cex=3)
text(npc[,1], npc[,2], c(1,2), col="white")
```



Q18. Which of these new patients should we prioritize for follow up based on your results?

Patient 2 falls into the range of red points (majority malignant cluster), so they should be prioritized.